UNITED STATES PATENT AND TRADEMARK OFFICE

UNITED STATES DEPARTMENT OF COMMERCE
United States Patent and Trademark Office
Address: COMMISSIONER FOR PATENTS
P.O. Box 1450
Alexandria, Virginia 22313-1450
www.uspto.gov

APPLICATION NO.	FILING DATE	FIRST NAMED INVENTOR	ATTORNEY DOCKET NO.	CONFIRMATION NO.
10/681,556	10/08/2003	Henry Chang	100201439-1	7781

22879 7590 10/16/2006

HEWLETT PACKARD COMPANY
P O BOX 272400, 3404 E. HARMONY ROAD
INTELLECTUAL PROPERTY ADMINISTRATION
FORT COLLINS, CO 80527-2400

EXAMINER	
WANG, BEN C	

ART UNIT	PAPER NUMBER
2196	

DATE MAILED: 10/16/2006

Please find below and/or attached an Office communication concerning this application or proceeding.

Office Action Summary	Application No.	Applicant(s)	
	10/681,556	CHANG ET AL.	
	Examiner	Art Unit	
	Ben C. Wang	2196	

-- The MAILING DATE of this communication appears on the cover sheet with the correspondence address --

Period for Reply

A SHORTENED STATUTORY PERIOD FOR REPLY IS SET TO EXPIRE 3 MONTH(S) OR THIRTY (30) DAYS, WHICHEVER IS LONGER, FROM THE MAILING DATE OF THIS COMMUNICATION.

- Extensions of time may be available under the provisions of 37 CFR 1.136(a). In no event, however, may a reply be timely filed after SIX (6) MONTHS from the mailing date of this communication.
- If NO period for reply is specified above, the maximum statutory period will apply and will expire SIX (6) MONTHS from the mailing date of this communication.
- Failure to reply within the set or extended period for reply will, by statute, cause the application to become ABANDONED (35 U.S.C. § 133). Any reply received by the Office later than three months after the mailing date of this communication, even if timely filed, may reduce any earned patent term adjustment. See 37 CFR 1.704(b).

Status

1)☒ Responsive to communication(s) filed on *08 October 2003*.

2a)☐ This action is **FINAL**. 2b)☒ This action is non-final.

3)☐ Since this application is in condition for allowance except for formal matters, prosecution as to the merits is closed in accordance with the practice under *Ex parte Quayle*, 1935 C.D. 11, 453 O.G. 213.

Disposition of Claims

4)☒ Claim(s) *1-17* is/are pending in the application.

4a) Of the above claim(s) _____ is/are withdrawn from consideration.

5)☐ Claim(s) _____ is/are allowed.

6)☒ Claim(s) *1-6,8-13 and 15-17* is/are rejected.

7)☒ Claim(s) *7 and 14* is/are objected to.

8)☐ Claim(s) _____ are subject to restriction and/or election requirement.

Application Papers

9)☐ The specification is objected to by the Examiner.

10)☐ The drawing(s) filed on _____ is/are: a)☐ accepted or b)☐ objected to by the Examiner.

Applicant may not request that any objection to the drawing(s) be held in abeyance. See 37 CFR 1.85(a).

Replacement drawing sheet(s) including the correction is required if the drawing(s) is objected to. See 37 CFR 1.121(d).

11)☐ The oath or declaration is objected to by the Examiner. Note the attached Office Action or form PTO-152.

Priority under 35 U.S.C. § 119

12)☐ Acknowledgment is made of a claim for foreign priority under 35 U.S.C. § 119(a)-(d) or (f).

a)☐ All b)☐ Some * c)☐ None of:

1.☐ Certified copies of the priority documents have been received.

2.☐ Certified copies of the priority documents have been received in Application No. _____.

3.☐ Copies of the certified copies of the priority documents have been received in this National Stage application from the International Bureau (PCT Rule 17.2(a)).

* See the attached detailed Office action for a list of the certified copies not received.

Attachment(s)

1) ☒ Notice of References Cited (PTO-892)
2) ☐ Notice of Draftsperson's Patent Drawing Review (PTO-948)
3) ☐ Information Disclosure Statement(s) (PTO/SB/08) Paper No(s)/Mail Date _____.
4) ☐ Interview Summary (PTO-413) Paper No(s)/Mail Date. _____.
5) ☐ Notice of Informal Patent Application
6) ☐ Other: _____.

DETAILED ACTION

1. Claims 1-17 are pending in this application and presented for examination.

Claims Objection

2. Claims 7 and 14 are objected to because the following phrases lack antecedent basis:

a) "each public function", claim 7, line 10 and claim 14 line 10.

b) "all remaining public and private functions", claim 7, lines 15-16 and claim 14, lines 15-16.

Claim Rejections – 35 USC § 101

3. 35 U.S.C. 101 reads as follows:

> Whoever invents or discovers any new and useful process, machine, manufacture, or composition of matter, or any new and useful improvement thereof, may obtain a patent therefor, subject to the conditions and requirements of this title.

4. Claims 10 and 15 are rejected under 35 U.S.C 101 because the claims are directed to non-statutory subject matter.

5. **As to claim 10**, the claim recites "software having machine readable code", lines 2-3. The software appears to be a functional descriptive material not embedded in machine-readable media. The claim also recites "for examining an application software program and producing a result", lines 4-5 and 11-12; however, it appears that

applicant's specification provides no explicit and deliberate definition of "examining an application software" and "producing a result".

6. **As to claim 15**, the claim recites software that appears to be a functional descriptive material not embedded in machine-readable media.

Claim Rejections – 35 USC § 112

7. The following is a quotation of the second paragraph of 35 U.S.C. 112:

> The specification shall conclude with one or more claims particularly pointing out and distinctly claiming the subject matter which the applicant regards as his invention.

8. Claims 1-17 are rejected under 35 U.S.C. 112, second paragraph, as being indefinite for failing to particularly point out and distinctly claim the subject matter which applicant regards as the invention.

9. **Claim 1** recites the limitation "examining an application software program" in line 2. It is not clearly understood what are the specifics, the purpose, and the goals of the examining process.

10. **Claim 10** recites the limitation "software for testing object-oriented system software" in lines 1-2. Applicant fails to particularly point out and distinctly claim the subject matter presented in the specification by using the term "software" instead of "a software".

11. The following terms or phrases are not clearly understood, rendering the corresponding claims vague and indefinite:

12. **Claim 15** recites the limitations "means for examining an application software program", lines 2-3. It is not clearly understood what are the specifics; the purpose, and the goals of the examination process.

13. **Claim 17** recites the limitation "examining said application software program", line 6. It is not clearly understood what are the specifics; the purpose, and the goals of the examination process.

14. **Claims 2-9, 11-14 and 16** are rejected as they depended from rejected claims.

Claim Rejections – 35 USC § 103(a)

15. The following is a quotation of 35 U.S.C. 103(a) which forms the basis for all obviousness rejections set forth in this Office action:

> (a) A patent may not be obtained though the invention is not identically disclosed or described as set forth in section 102 of this title, if the differences between the subject matter sought to be patented and the prior art are such that the subject matter as a whole would have been obvious at the time the invention was made to a person having ordinary skill in the art to which said subject matter pertains. Patentability shall not be negatived by the manner in which the invention was made.

16. Claims 1, 4, 5, 8-12, 15 and 17 are rejected under 35 U.S.C. 103(a) as being unpatentable over Shmuel Ur et al. (hereafter 'Shmuel Ur') (US 2003/0110474 A1) in view of Cahill et al. (hereafter 'Cahill'), *'The Java Metrics Reporter – An Extensible Tool*

for OO Software Analysis', 2002 IEEE and further in view of Benlarbi et al. (hereafter 'Benlarbi'), *'Polymorphism Measures for Early Risk Prediction'*, 1999, ACM.

17. **As to claim 1**, Shmuel Ur discloses that a method for testing software, comprising: examining an application software program including calls to both a static analysis tool and a dynamic analysis tool ([0030], lines 1-7), but Shmuel Ur does not explicitly disclose calls to system classes. However, in an analogous art, Cahill discloses calls to system classes (Sec. 2.3, the table on page 512, the 1st entry – System, Root Classes). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of both Shmuel Ur and Cahill in order to call system classes and identify them as the dominating blocks in Shmuel Ur's system. The motivation is to put highly used software into system classes that can be shared between various application classes without mixing them with application code. Shmuel Ur does not disclose that determining a static use count and deriving dynamic use count and further assigning a proportional weighting attribute to them. However, in an analogous art, Cahill discloses that determining a static use count of said system classes; deriving a dynamic use count of each of said system classes during operation of said application software program; assigning a proportional weighing attribute to each system class based on its corresponding static use count and dynamic use count; (Sec. 3.1 Present Selection and Rationale, 1st paragraph – fifteen metrics organized into the categories Basic (Sec. 3.2), Complexity (3.3), Inheritance (Sec. 3.4) and Polymorphism (Sec.3.5). Sec. 3.4, Method Inheritance Factor (MIF), lines

1-5 for static use count portion; Sec. 3.5, Polymorphism Factor (PF), lines 5-11 for dynamic/static use count portion). It is also well know in the art that polymorphism metrics measure include both static and dynamic information (see Benlarbi, Sec. 3.1 Forms of Polymorphism in OO design, 5th paragraph – classification of polymorphic behaviors includes polymorphic behaviors that are based on run-time binding (dynamic) decisions as well as on compile time (static) linking decision). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Shmuel Ur, Cahill, and Benlarbi in order to collect both static and dynamic use count and assigning proportional weighing attributes. The motivation is that inheritance factor metric (for static use count) is useful in providing a direct measure of inheritance in a system, and from this, the level or reuse in the system, and polymorphism factor metric (for dynamic count use) is useful in indicating complexity, comprehensibility and maintainability. Shmuel Ur discloses that testing said 'system classes' (the dominating blocks) in order according to said corresponding proportional weighing attributes ([0005], lines 12-15; [0006], lines 22-26 and [0007]) Shmuel Ur discloses a business model factors for testing software ([0004], lines 12-14).

18. **As to claim 10,** a software for testing object-oriented system software having system classes, the software having machine readable code for performing the step: comprising: examining an application software program including running both a static analysis tool and a dynamic analysis tool ([0030], lines 1-7), but Shmuel Ur does not explicitly disclose calls to system classes. However, in an analogous art, Cahill

discloses calls to system classes (Sec. 2.3, the table on page 512, the 1st entry – System, Root Classes). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of both Shmuel Ur and Cahill in order to call to system classes. The motivation is to put highly used software into system classes that can be shared between various application classes without mixing them with application code. Shmuel Ur does not disclose that determining a static use count and deriving dynamic use count and further assigning a proportional weighting attribute to them. However, in an analogous art, Cahill discloses that determining a static use count of said system classes; deriving a dynamic use count of each of said system classes during operation of said application software program; assigning a proportional weighing attribute to each system class based on its corresponding static use count and dynamic use count; (Sec. 3.1 Present Selection and Rationale, 1st paragraph – fifteen metrics organized into the categories Basic (Sec. 3.2), Complexity (3.3), Inheritance (Sec. 3.4) and Polymorphism (Sec.3.5). Sec. 3.4, Method Inheritance Factor (MIF), lines 1-5 for static use count portion; Sec. 3.5, Polymorphism Factor (PF), lines 5-11 for dynamic/static use count portion). It is also well know in the art that polymorphism metrics measure include both static and dynamic information (see Benlarbi, Sec. 3.1 Forms of Polymorphism in OO design, 5th paragraph – classification of polymorphic behaviors includes polymorphic behaviors that are based on run-time binding (dynamic) decisions as well as on compile time (static) linking decision). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Shmuel Ur, Cahill in and Benlarbi order

to collect both static and dynamic use count and assigning proportional weighing attributes. The motivation is that inheritance factor metric (for static use count) is useful in providing a direct measure of inheritance in a system, and from this, the level or reuse in the system, and polymorphism factor metric (for dynamic count use) is useful in indicating complexity, comprehensibility and maintainability. Shmuel Ur discloses that testing said 'system classes' (the dominating blocks) in order according to said corresponding proportional weighing attributes ([0005], lines 12-15; [0006], lines 22-26 and [0007]) Shmuel Ur discloses a business model factors for testing software ([0004], lines 12-14).

19. **As to claim 15**, a software tester, comprising: means for examining an application software program including calls to both a static analysis tool and a dynamic analysis tool ([0030], lines 1-7), but Shmuel Ur does not explicitly disclose to call to system classes. However, in the analogous art, Cahill discloses to call to system classes (Sec. 2.3, the table on page 512, the 1st entry – System, Root Classes). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of both Shmuel Ur and Cahill in order to call to system classes. The motivation is to put highly used software into system classes that can be shared between various application classes without mixing them with application code. Shmuel Ur does not disclose that determining a static use count and deriving dynamic use count and further assigning a proportional weighting attribute to them. However, in an analogous art, Cahill discloses that determining a static use

count of said system classes; deriving a dynamic use count of each of said system classes during operation of said application software program; assigning a proportional weighing attribute to each system class based on its corresponding static use count and dynamic use count; (Sec. 3.1 Present Selection and Rationale, 1st paragraph – fifteen metrics organized into the categories Basic (Sec. 3.2), Complexity (3.3), Inheritance (Sec. 3.4) and Polymorphism (Sec.3.5). Sec. 3.4, Method Inheritance Factor (MIF), lines 1-5 for static use count portion; Sec. 3.5, Polymorphism Factor (PF), lines 5-11 for dynamic/static use count portion). It is also well know in the art that polymorphism metrics measure include both static and dynamic information (see Benlarbi, Sec. 3.1 Forms of Polymorphism in OO design, 5th paragraph – classification of polymorphic behaviors includes polymorphic behaviors that are based on run-time binding (dynamic) decisions as well as on compile time (static) linking decision). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Shmuel Ur, Cahill in and Benlarbi order to collect both static and dynamic use count and assigning proportional weighing attributes. The motivation is that inheritance factor metric (for static use count) is useful in providing a direct measure of inheritance in a system, and from this, the level or reuse in the system, and polymorphism factor metric (for dynamic count use) is useful in indicating complexity, comprehensibility and maintainability. Shmuel Ur discloses that testing said 'system classes' (the dominating blocks) in order according to said corresponding proportional weighing attributes ([0005], lines 12-15; [0006], lines 22-26 and [0007]) Shmuel Ur discloses a business model factors for testing software ([0004], lines 12-14).

20. **As to claim 17**, a business model for testing software, comprising: setting a resource limit on the available time or money that is devoted to testing a particular application software program; stopping testing when said resource limit is reached ([0097], lines 1-6; [0049]) examining an application software program including calls to both a static analysis tool and a dynamic analysis tool ([0030], lines 1-7), but Shmuel Ur does not explicitly disclose to call to system classes. However, in the analogous art, Cahill discloses to call to system classes (Sec. 2.3, the table on page 512, the 1st entry – System, Root Classes). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of both Shmuel Ur and Cahill in order to call to system classes. The motivation is to put highly used software into system classes that can be shared between various application classes without mixing them with application code. Shmuel Ur does not disclose that determining a static use count and deriving dynamic use count and further assigning a proportional weighting attribute to them. However, in an analogous art, Cahill discloses that determining a static use count of said system classes; deriving a dynamic use count of each of said system classes during operation of said application software program; assigning a proportional weighing attribute to each system class based on its corresponding static use count and dynamic use count; (Sec. 3.1 Present Selection and Rationale, 1st paragraph – fifteen metrics organized into the categories Basic (Sec. 3.2), Complexity (3.3), Inheritance (Sec. 3.4) and Polymorphism (Sec.3.5). Sec. 3.4, Method Inheritance Factor (MIF), lines 1-5 for static use count portion; Sec. 3.5, Polymorphism

Factor (PF), lines 5-11 for dynamic/static use count portion) It is also well know in the art that polymorphism metrics measure include both static and dynamic information (see Benlarbi, Sec. 3.1 Forms of Polymorphism in OO design, 5th paragraph – classification of polymorphic behaviors includes polymorphic behaviors that are based on run-time binding (dynamic) decisions as well as on compile time (static) linking decision). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of Shmuel Ur, Cahill in and Benlarbi order to collect both static and dynamic use count and assigning proportional weighing attributes. The motivation is that inheritance factor metric (for static use count) is useful in providing a direct measure of inheritance in a system, and from this, the level or reuse in the system, and polymorphism factor metric (for dynamic count use) is useful in indicating complexity, comprehensibility and maintainability. Shmuel Ur discloses that testing said 'system classes' (the dominating blocks) in order according to said corresponding proportional weighing attributes ([0005], lines 12-15; [0006], lines 22-26 and [0007]) Shmuel Ur discloses a business model factors for testing software ([0004], lines 12-14).

21. **As to claims 4 and 11**, Shmuel Ur does not specifically disclose that producing a static use count further comprises assigning a static observation percentage to each system class by dividing said static use count by a sum of all static use counts. However, in an analogous art, Cahill discloses that producing a static use count further comprises assigning a static observation percentage to each system class by dividing

said static use count by a sum of all static use counts (Sec. 3.4, Method Inheritance Factor (MIF), lines 2-5). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of both Shmuel Ur and Cahill in order to produce static observation percentage for each system class. The motivation is that static observation percentage is useful in providing a direct measure of inheritance in a system, and from this, the level of reuse in the system.

22. **As to claims 5 and 12**, Shmuel Ur does not specifically disclose that producing a dynamic use count further comprises assigning a dynamic observation percentage to each system class by dividing said dynamic use count by a sum of all dynamic use counts. However, in an analogous art, Cahill discloses that producing a dynamic use count further comprises assigning a dynamic observation percentage to each system class by dividing said dynamic use count by a sum of all dynamic use counts (Sec. 3.5, Polymorphism Factor (PF), Lines 2-7). Therefore, it would have been obvious to one of ordinary skill in the art at the time the invention was made to combine the teachings of both Shmuel Ur and Cahill in order to produce dynamic observation percentage for each system class. The motivation is that dynamic observation percentage is useful in indicating complexity, comprehensibility and maintainability.

23. **As to claim 8**, Shmuel Ur discloses that the testing system classes further comprises ending a test when a testing resource is exhausted and prior to testing a last entry have a least weight ([0004], lines 12-14).

24. **As to claims 9**, Shmuel Ur discloses that the testing system classes further comprises ending a test when at least a limit of available time or funding is exhausted and prior to testing a last entry having a least weight ([0004], lines 12-14).

25. Claims 6 and 13 are rejected under 35 U.S.C. 103(a) as being unpatentable over Shmuel Ur, in view of Cahill & Benlarbi and further in view of T. Ball (hereafter 'Ball'), *'The Concept of Dynamic Analysis'*, Bell Laboratories Lucent Technologies, Oct, 1999.

26. **As to claims 6 and 13**, Shmuel Ur or Benlarbi or Cahill does not disclose that producing a static use count further comprises assigning a static observation percentage to each system class by diving the static use count by a sum of all static use counts; and producing a dynamic use count further comprises assigning a dynamic observation percentage to each system class by dividing the dynamic use count by a sum of all dynamic use counts. However, in an analogous art, Ball discloses that producing a static use count further comprises assigning a static observation percentage to each system class by diving the static use count by a sum of all static use counts; and producing a dynamic use count further comprises assigning a dynamic observation percentage to each system class by dividing the dynamic use count by a sum of all dynamic use counts. (Sec. 1, dynamic and static analyses are complementary techniques in a number of dimensions: a) completeness, b) scope, and c) precision). Therefore, it would have been obvious to one of ordinary skill in the art at

the time the invention was made to combine the teachings of Shmuel Ur, Cahill, Benlarbi and Ball in order to produce static plus dynamic observation percentages for each system class. The motivation is that both static and dynamic observation percentages for each system class are complementary metrics factors to product more complete, precise, scope combined software metrics report.

27. Claims 2-3 and 16 are rejected under 35 U.S.C. 103(a) as being unpatentable over Shmuel Ur, Cahill and Benlarbi, in view of Kuzmin et al. (hereafter 'Kuzmin'), (US 7,039,902 B2)

28. **As to claims 2, 16**, Shmuel Ur, Cahill and Benlarbi do not disclose that only the most heavily weighted portion of all such system classes are test at all. However, in an analogous art, Kuzmin discloses that only the most heavily weighted portion of all such system classes are tested at all (Col. 5, lines 48, 53). It would have been obvious to one of ordinary skill in the art at the time of invention was made to combine the teachings of Shmuel Ur, Cahill, Benlarbi and Kuzmin in order to focus on testing in most heavily weighted portion of all such system classes. The motivation is that prioritization of untested portions of code is advantageous because it finds the untested portions with the most potential impact in the body of application code.

29. **As to claims 3,** Shmuel Ur discloses that composing and executing formal rules are required (Fig. 6, step 148, step 149, step 152 and step 156). But Shmuel Ur, Cahill

and Benlarbi do not specifically disclose that only those system classes that are actually used in operation of said application software program are tested at all. However, in an analogous art, Kuzmin discloses that only those system classes that are actually used in operation of said application software program are tested at all (Col. 2, lines 1-10). It would have been obvious to one of ordinary skill in the art at the time of invention was made to combine the teachings of Shmuel Ur, Cahill, Benlarbi and Kuzmin in order to test only used system classes. The motivation is to save costs and avoid any delays and make earlier software product delivery.

Allowable Subject Matter

30. Claims 7 and 14 are objected to as being dependent upon a rejected base claim, but would be allowable if rewritten to overcome the rejections under 35 U.S.C. 101 and 35 U.S.C. 112, 2nd paragraph set forth in this office action and to include all the limitations of the base claim and any intervening claims.

31. The following is an examiner's statement of reasons for allowance:

Regarding claims 7 and 14, prior art of record fails to reasonably show or suggest the specific weighting scheme as claimed. Specifically, the assigning a first weight defined by a first constant plus a sum of the static use count plus the dynamic use count, a second weight equal to the first constant, further assigning a third weight as a second constant that is less than the first constant (which is added a sum of the

static and dynamic observation percentage), and assigning to all remaining classes a fourth weight less than the second constant.

Any comments considered necessary by applicant must be submitted no later than the payment of the issue fee and, to avoid processing delays, should preferably accompany the issue fee. Such submissions should be clearly labeled "Comments on Statement of Reasons for Allowance."

Conclusion

Any inquiry concerning this communication or earlier communications from the examiner should be directed to Ben C. Wang whose telephone number is 571-270-1240. The examiner can normally be reached on Monday - Friday, 8:00 a.m. - 5:00 p.m., EST.

If attempts to reach the examiner by telephone are unsuccessful, the examiner's supervisor, Nabil El-Hady can be reached on 571-272-2333. The fax phone number for the organization where this application or proceeding is assigned is 571-273-8300.

Information regarding the status of an application may be obtained from the Patent Application Information Retrieval (PAIR) system. Status information for published applications may be obtained from either Private PAIR or Public PAIR. Status information for unpublished applications is available through Private PAIR only. For more information about the PAIR system, see http://pair-direct.uspto.gov. Should you have questions on access to the Private PAIR system, contact the Electronic Business Center (EBC) at 866-217-9197 (toll-free). If you would like assistance from a USPTO Customer Service Representative or access to the automated information system, call 800-786-9199 (IN USA OR CANADA) or 571-272-1000.

BCW

September 18, 2006

NABIL M. EL-HADY
SUPERVISORY PATENT EXAMINER